Color descriptors for Web image retrieval: a comparative study

Otávio Augusto Bizetto Penatti and Ricardo da Silva Torres
otavio@lis.ic.unicamp.br, rtorres@ic.unicamp.br
Institute of Computing, University of Campinas - UNICAMP
13084-970, Campinas, SP, Brazil

Abstract

This paper presents a comparative study of color descriptors for content-based image retrieval on the Web. Several image descriptors were compared theoretically and the most relevant ones were implemented and tested in two different databases. The main goal was to find out the best descriptors for Web image retrieval. Descriptors are compared according to the extraction and distance functions complexities, the compactness of feature vectors, and the ability to retrieve relevant images.

1 Introduction

In recent years, the popularization of digital image and video devices has made image collections' growth become exponential. In fact, a huge amount of visual information is available to a large number of users. In this scenario, there is a demand for searching systems able to easily, efficiently, and effectively retrieve this kind of information.

The most used approach for visual information retrieval nowadays is based on text annotations. Images are associated with keywords and these keywords are used for indexing and searching for images. The main advantage of this approach is that it relies on semantically enriched descriptions. However, the process of describing images textually usually leads to interpretation problems. Different people may have different interpretation of the same image.

Another approach to retrieve visual information is the so-called content-based image retrieval (CBIR). This approach describes images using low-level features, like color, texture or shape of objects. The main advantage of this approach is that it does not suffer from the subjectiveness of textual description. Its main drawback is related with the lack of semantic information, also called semantic gap.

CBIR is based on image descriptors. They are responsible for extracting image features like color, texture or shape of objects. The features extracted are used for image comparison.

Color is the visual feature most used in CBIR systems and also the most explored in literature. The main reason is because humans tend to differentiate images mostly by means of color features. Despite the large number of color image descriptors in literature, most of them are not suitable for heterogeneous databases with a huge number of images. One problem with many proposed color descriptors is that their time and space complexities are usually not analyzed. Furthermore, when proposed, they are tested in restricted image databases. For example, these databases have only images from a specific domain.

This paper presents a comparative study of color image descriptors. The evaluated set of descriptors includes widely used and recently proposed in the literature. The main focus is to evaluate their use on the Web, a general purpose application with a huge amount and heterogeneous visual content.This paper may serve as a guide to developers of image search engines.

The paper is organized as follows. The next section contains an overview about color description. Section 3 shows the main issues related to Web image retrieval. Section 4 presents the evaluation criteria for this comparative study of color descriptors. Section 5 contains the results of our comparative study, giving as brief description of the descriptors chosen for our experiments and the reason for the choice based on the criteria previously presented. Section 6 shows our experiments involving a large and heterogeneous image database. Concluding remarks are presented in Section 7.

2 Color description overview

One of the most important features visually recognized by humans in images is color. Humans tend to distinguish images based mostly on color features. Because of this, color features are the most used in CBIR systems and the most studied in literature.

Formally, a descriptor D can be defined as a pair (ϵ_D, δ_D) [3]. ϵ_D is an algorithm for extracting image features, which are stored in *feature vectors*. The δ_D function is responsible for comparing these vectors. A distance (or

similarity) between two vectors is calculated by δ_D. This distance is assumed to be the distance between the images the vectors represent. Usually, the similarity is the inverse of the distance. Most of the descriptors analyze just one kind of image feature. The descriptors studied in this work consider color information, because of this they are called color descriptors.

There are three main approaches for color description.

One approach considers the color image information globally. This approach is usually simple, because no preprocessing or segmentation step is necessary to extract features, what leads to fast feature extraction. However, it has some drawbacks. For example, in many cases, spatial distribution of colors are not encoded resulting in low effectiveness. Most descriptors of this category codifies image features in histograms. The most common one is the Global Color Histogram [23].

Another approach partitions the image into fixed-size cells and extracts information from each cell. This approach usually codifies more spatial information, however it can lead to high space overhead. A descriptor of this kind is the Local Color Histogram [23].

A third approach divides the image into regions, which may be different to each image. The division is often made by a segmentation algorithm. Therefore, it introduces a step that increases the complexity of the extraction process. In general, this approach presents higher effectiveness retrieval, however it has higher computational cost. A descriptor of this category is Color Coherence Vector [17].

The most desirable characteristics of an image descriptor are: fast feature extraction, compact feature vectors generation, fast distance calculation, and high effectiveness.

3 Web: a general purpose application

Recently, Web is becoming more and more important, being present in everyday life of millions of people. It contains a very huge amount of content about every kind of subject. Because of this, Web is nowadays one of the largest and most heterogeneous databases in the world. A great amount of this database is composed by visual elements, like images and videos. In 2003 [10], there were 425 million images indexed by Google Image Search. This leads to a big challenge: how to retrieve this visual information.

As presented in previous sections, CBIR systems with color descriptors can be used for this task. The common searching process is characterized as follows. A query is defined by a user (for example, by defining a query image) and the system has to find images relevant to the query. This is done by extracting query image features and comparing them to the features of the database images. Each comparison returns a distance between the images and this distance is used to rank the database images. This will define which of them will be shown to the user.

When a search engine is used in the Web, the user cannot wait too much to see the results. Otherwise, the search engine will not be used anymore. For example, consider searching for an image in the Web. In this case, a huge number of images must be compared to the query. Because of this, descriptors must be very fast to compare images and must also be fast to extract image features in order to give high efficiency for a Web CBIR system.

Descriptors should also generate compact feature vectors. The amount of space required to store feature vectors is proportional to the number of images in the database. The more compact the feature vectors are, the less is the impact of storage requirements in the CBIR system. As the Web has a huge number of images, the compacteness of feature vectors is even more important.

Another important issue related to Web image retrieval is the evaluation of effectiveness. Again, Web has characteristics that require attention in this process. Since database images are not known, it is not possible to classify them in categories or classes. This makes it impossible to automatically extract some evaluation measures from the results, and, in consequence, it makes the evaluation process slower. Also, as we do not know the whole image database, it is impossible to know if the best ranked images are the most relevant to the query.

One way to evaluate this kind of system is using real users. They can indicate which images are relevant and which images are not for a given query result. After this, evaluation measures can be computed. We consider that real users are a reliable way to test the system. They will reflect the same using conditions that the system will face when being used in real life.

4 Evaluation criteria

We compared several color image descriptors based on the main issues presented in previous sections. We choose recently proposed descriptors and descriptors that are important in literature. Next we define the main criteria considered in the comparison.

4.1 Feature extraction complexity

The feature vector extraction algorithm of a descriptor is used when the features of an image needs to be extracted. This occurs in two different moments in a CBIR system. The first moment is when the image is gathered from the Web and is inserted into the system database. At this time, its features need to be extracted and stored into the database. This is often done off line. The other moment occurs when a query is sent to the system (we are considering here query

by image example). If the query image is not an image from the system database, its features need to be extracted at running time. The system will be able to start the search only after the query image features are extracted.

Even considering that most of the extraction task is done off line, an important step in done on line. This shows that fast extraction functions are desirable.

4.2 Distance function complexity

The distance function of a descriptor, usually, is responsible for the major amount of computational time required for a query processing. During a search, the query image must be compared to many other images of the database, leading to a very large number of comparisons, or better, leading to a very large number of distance calculations. Considering a sequential scan, the query image must be compared to every other image in the database.

The distance function is also very important due to indexing requirements. Large databases often require the use of an indexing structure. Without it a query would take much time. Indexing structures reduce the number of image comparisons, consequently reducing the retrieval time. They eliminate comparisons with potentially dissimilar images. The indexing of descriptors distance functions is highly desirable. Metric distance functions, like L1 and L2, are example of indexable functions.

4.3 Space requirements

The features extracted by the extraction algorithm of a descriptor are usually stored into *feature vectors*. Every image in the database will have one feature vector per descriptor, in other words, the amount of space required to store feature vectors is proportional to the quantity of images in the database. Considering the Web scenario, where a huge number of images exists, a very large space will be required to store feature vectors. We may also consider that more than one descriptor should be used to deal with Web images heterogeneity. Therefore, this space is even a multiple of the number of descriptors used. This emphasizes the importance of generating compact feature vectors.

Many extraction algorithms generate constant size feature vectors, that do not depend on the size of the image. However, there are others that generate variable-size vectors.

4.4 Descriptor validation

We also considered in our evaluation the test environment used to validate a given descriptor. By test environment we mean the databases with which it was tested, to which other descriptors it was compared and the evaluation measures used. The experimental setup in a descriptor proposal could show us if the descriptor is able to lead with heterogeneous and large databases. This is important to determine if the descriptor can be used on the Web. The comparison of the given descriptor to other existing descriptors shows us that, if the descriptor is compared only with variations of itself, it is a reason to believe that it was not able to show advantages over the most important existing descriptors.

5 Evaluation results

We analyzed several color descriptors based on the papers in which they were proposed. We tried to find the most relevant information according to the criteria presented previously. The analysis result is summarized in Table 1 and Table 2. Table 1 shows the extraction algorithm and distance function complexities and the space requirements of each of the descriptors studied. Table 2 shows the main information about the descriptors' validation.

In some papers the information could be gathered directly, because they were explicitly presented. For example, some papers presented the descriptor extraction algorithm complexity in O notation. However, in the great majority of papers, we had to analyze deeply to obtain the information we were looking for, because they were not explicitly presented. Besides this, some papers do not present enough information to analyze some criteria.

Analyzing the tables we can see that descriptors like [1, 7, 14, 19, 27, 28] have extraction algorithms with an extra complexity related to the most common $O(n)$ complexity. The faster the extraction algorithm, the least the impact of this algorithm in a Web CBIR system. Descriptors like [1, 6, 9, 19] present a distance function with an additional complexity. This can make a Web CBIR system not so fast to response user requests. Descriptors like [6–9, 20, 22] generate little compact feature vectors what would require high storage capability. Considering Table 2 it is possible to see some descriptors that were tested in image databases with few images or with images from a specific domain, like the descriptors proposed in [2, 6, 15, 16]. Some descriptors were compared only with themselves like [2, 13, 25]. Although some descriptors are not good for some criteria, they can be very good in other ones. For example, the descriptor presented in [25] was compared only with itself, but its extraction algorithm generates very compact feature vectors. Another example is the descriptor proposed in [16]. Although its experiments had been run in a database with images from a specific domain, it has a very fast distance function.

Based on the comparison criteria, we choose some descriptors to implement and test in two image databases. We also choose descriptors that are usually used as base-

Table 1. Descriptors comparison: extraction and distance functions complexity, and space requirements. Caption: n is the number of pixels in an image, NC=not clear, ND=not mentioned.

Descriptor	Extraction algorithm	Distance function	Feature Vector Size
DCSD [27]	O(GLA)+O(n*Q)+O(n), GLA=GLA algorithm, Q=quantity of colors	O(Nd), Nd=quantity of dominant colors	O(Nd)
[28]	O(n)+O(SIFT), SIFT=SIFT algorithm	O(vector_size)	722 dimensions (without PCA) and 188 dimensions (with PCA)
CW-LUV [15]	O(n)	O(vector_size)	127 bits
CW-HSV [25]	O(n)	O(vector_size)	63 bits
[1]	O(SIFT)+O(Keys construction with color invariants gradients)	O(Nkeys^2), Nkeys=number of keypoints	O(Nkeys)
[8]	O(n)+O(nb), nb=number of bins in HLS histogram	O(Nc), Nc=number of colors in HLS space	(Quantity of border pixels * [integer]) + 2*nb
MPEG-7 [12]	MPEG-7 descriptors	several	several
[13]	O(n)	O(nb), nb=number of bins	256 bins
[21]	O(n)	O(nb), nb=number of bins	64 and 256 bins
[26]	O(n)	O(nb), nb=number of bins	*NC*
[18]	O(n)	O(nb), nb=number of bins	*NC*
[7]	O(n^3)	*NC*	(number of pixels)*(fi=L,a,b,x,y).
[14]	O(n^2)	O(nb), nb=number of bins	96 float values
[20]	O(n)	O(nb), nb=number of bins	1856 bins (on average) and 4096 bins (at most)
[11]	O(n)	O(nb), nb=number of bins	3*m + 6 float values, m=quantity of blocks
CDE [22]	O(n)+O(N)+O(nb), N=quantity of concentric circles, nb=number of bins	O(nb), nb=number of bins	(h1, E1, H, hi, Ei, H, hn, En), where i is the bin (n bins)
[2]	*NC*	Vector Angular Distance	nc*(Pi,Li), nc=number of color regions, Pi=percentage of region i, Li=location of region i
[6]	O(n*y), y=gaussian filter	O(t(L-1)), t=quantization levels, L=quantity of resolutions	H = [h0,h1,...,hL-1], hi=histogram of resolution i
[4]	O(n)	O(nb), nb=number of bins	128 float values
[9]	O(nb*tb), nb=number of blocks, tb=block size	O(k*m), k=query image blocks(k $\times$ k), m=database images blocks (m $\times$ m)	Value or Histogram for each block. Worst case: O(nbins*m^2) nbins=number of bins
CM [16]	O(n) + O(Q^2*QM), Q=quantization levels, QM=quantity of moments	O(QM)	QM*2 integer values
BIC [5]	O(n)	O(nb), nb=number of bins	128 bins
CBC [19]	O(nlogn)	O(nlogn)	Q*6 float values, Q=quantity of regions
CCV [17]	O(n)	O(Q), Q=quantization levels	Q*2 integer values
GCH [23]	O(n)	O(nb), nb=number of bins	64 bins
LCH [23]	O(n)	O(nb), nb=number of bins	1024 bins (4x4 cells)

lines for comparisons. A full description about the descriptors is beyond the scope of this paper. Next we show a brief description of the chosen ones: Global Color Histogram (GCH), Local Color Histogram (LCH), Color Coherence Vector (CCV), Border/Interior Pixel Classification (BIC), Chromaticity Moments (CM), and Color Wavelet (CW-HSV).

The Global Color Histogram (GCH) [23] is the color descriptor most cited in literature and is often used as a baseline for descriptors comparison. It analyzes the whole color information of the image. Usually, a quantization step is required to reduce the number of distinct colors. In our experiments the 64 bin color histogram was used.

The Local Color Histogram (LCH) [23] is a partition-based descriptor which divides the image in fixed size cells and calculates a color histogram of each cell. The distance function compares the corresponding cell histograms. In our experiments, LCH generated feature vectors with 1024 bins and L1 distance function was used.

The Color Coherence Vector (CCV) descriptor [17] classifies each pixel in either coherent or incoherent, based on whether or not it is part of a large similarly-colored region. The CCV extraction algorithm first blurs the image and the color space is discretized to eliminate small variations between neighbor pixels. Next, the connected components of the image are found in order to classify the pixels in coherent or incoherent. Its distance function can be any that compare histograms. In our experiments, L1 distance was used.

GCH, LCH, and CCV were chosen because they are popular descriptors and each of them follows a different color extraction approach. GCH is global-based, LCH is partition-based and CCV is region-based.

BIC [5] is a region-based color descriptor, whose name comes from Border/Interior Pixel Classification. Its extraction algorithm classifies the image pixels in border or interior pixels. The image is first quantized into 64 colors in RGB color space. Then, each pixel is classified as *interior* if its neighbors (above, below, left and right pixels) have the same color. Otherwise it is classified as *border* pixel. After the classification, two histograms are generated: one for border pixels and other for interior pixels. These histograms are stored as one single histogram with 128 bins. The distance function is called *dLog* and it compares histograms in a logarithmic scale. The reasons for us to choose BIC are its good effectiveness in heterogeneous databases and its fast distance function.

The CM [16] descriptor characterizes images by chromaticity values. These values are calculated after image conversion to CIE *XYZ* color space. Chromaticity values *(x,y)* are then calculated as $x = \frac{X}{X+Y+Z}$ and $y = \frac{Y}{X+Y+Z}$. From these values, it is generated the chromaticities *trace*, which indicates the presence or not of a value *(x,y)* in the image. As it is possible to exist more than one image pixel with the same chromaticity value *(x,y)*, the chromaticities *histogram* is also calculated. The trace and histogram are used to define the chromaticity moments. In our experiments, like in [16], 6 moments were used. This results in a feature vector with 12 values. The distance function is very simple. It calculates the modular difference between corresponding moments. The reasons for us to choose CM are its compact feature vector generation and its fast distance function.

Table 2. Descriptors comparison: tests environment and validation. Caption: HET=Heterogeneous, HET-WEB=Heterogeneous and from the Web, NU=Not used, NC=No comparison with other descriptors, IT=Compared with variations of itself, ND=Not mentioned.

Descriptor	DB features	DB size	Evaluation measures	Comparisons
DCSD [27]	MPEG7 CCD	5466	ANMRR, NMRR	DCD, CSD, SCD, CLD
[28]	INRIA dataset	*ND*	Recall x (1-Precision)	SIFT, *IT*
CW-LUV [15]	*ND*	100 numbers	ANMRR, AVR, MRR, NMRR	HSV, CIELuv color spaces
CW-HSV [25]	*HETWEB*	2997	ANMRR	*IT*
[1]	ALOI (Amsterdam)	*ND*	number of repeated CSIFT features divided by number of SIFT features	SIFT
[8]	*HET*	5000	ANMRR, Precision, Recall	GCH, Hybrid Graph Representation, Color Correlogram
MPEG-7 [12]	CCD MPEG-7	5000	ANMRR	*IT*
[13]	*HET*	5400	ANMRR	*IT*
[21]	*HET* (185x123 pixels)	3000	Rank	GCH
[26]	Berkeley Digital Library	24000	Scope × Recall	GCH, Joint Histogram, and Color Correlogram
[18]	Cricket Low DOF	4986	Precision × Recall	Blobworld
[7]	(1) Manuscripts and (2) CLIC	(1) 1500 and (2) 15000	*NU*	Blobs+EMD segmentation
[24]	(1)COREL, (2)MPEG7, (3)MATTON	(1)1000, (2)5466, (3)126604	ANMRR	GCH, DCD, KLT QBIC, KLT (4 variations)
[14]	COREL	1000	Precision, Rank	Simplicity, WBIIS, Color Correlogram, Edge Correlogram
[20]	COREL	20000	Precision × Recall, $\theta_{abs}, \theta_{rel}$	GCH, CCV, Grid
[11]	(1) animations, (2) and (3) full color	(1) 800, (2) 470, (3) 10000	Retrieval accuracy	GCH, Color moments, and Chang and Liu's method
CDE [22]	*HETWEB*	8000	ANMRR, Recall, Precision	SCH, Geostat
[2]	COREL (architecture)	220	Precision, recall	*IT*
[6]	(1) Synthetic, (2) Brodatz, (3) CureT	(1) 108, (2) 91, (3) 8046	% texture matches	Fourier Power Spectrum, Color Features, Wavelet Packets, Co-ocurrency matrix, Markov random fields
[4]	*HET*	7000	Precision, Recall, Rank1, Rank$_\tau$ P_{20}, P_{50}, P_{N_R}, $R_P 0.5$, R_{100}	GCH, CCV
[9]	*ND*	*ND*	*ND*	GCH
CM [16]	VisTex (blur, noise, etc)	2266	Precision and Recall	Histogram Intersection
BIC [5]	COREL	20000	Precision × Recall, θ × Recall, P_R, P_{30}, R_{30}, P_{100}, R_{100}, 3P-Precision, 11P-Precision	GCH, CCV, CBC, Grid9
CBC [19]	COREL	20000	Precision × Recall, NavgR	CCV, CMM, GCH, Grid, CSH
CCV [17]	(1)Chabot, (2)QBIC, (3)COREL, (4)PhotoCD, (5)Videos scenes	(1)11667, (2)1440, (3)1005, (4)93, (5)349	Rank	GCH
GCH [23]	Objects in close	*ND*	Average Match Percentile	Histogram Intersection, Incremental Intersection
LCH [23]	Objects in close	*ND*	Average Match Percentile	Histogram Intersection, Incremental Intersection

The descriptor proposed in [25], here called just by CW-HSV, computes color features in wavelet domain. First it generates the image global color histogram in HSV color space. Then Haar transform coefficients of the histogram are calculated hierarchically by using Haar wavelet function. This is done dividing recursively the histogram in halves. The first level, the 64 bins of histogram is devided into two halves. If the sum of the histogram bins in the left half is greater than the sum of the histogram bins in the right half, the first bit of the feature vector is 1, else it is 0. The second level divides the two halves above into the middle and the same idea is applied. In the end, 63 binary values will compose the feature vector. The distance between two feature vectors is calculated by the Hamming distance. The reasons for us to choose CW-HSV are its compact feature vector generation (only 63 bits) and its fast distance function.

6 Experiments

The descriptors chosen were coded in C and tested in two image databases. The first database is the *eth-cropped-close* database (*eth*) with 3280 color images of one single object, like tomatoes, cars, and cups, for example. The objects appear in many variations of rotation. For instance, a car was photographed from different angles. This database is equally divided into 8 classes where each class represents a different object, and all images are 128x128 pixels.

The other database contains more than 250 thousand images from the Web that were gathered from the Yahoo! directory. It is a very heterogeneous database that represents well the content of the Web. There is no categorization or subdivision in classes and the images are stored the same way they were gathered, with no post-processing or dimension reduction.

The experiments were run in a computer with 2 Xeon Quadcore processors and 4GB of RAM. Despite the computer multi core capability, the implementation of the descriptors were not tunned to parallelization.

We used *eth* database to evaluate extraction and distance times, space requirements and effectiveness. Effectiveness was evaluated by precision × recall curves. We used all database images as query images. For *yahoo* database we analyzed effectiveness using only a part of the database. Two thousand images were collected from the full *yahoo* database. Each of the two thousand images was used as query image. In Section 6.2 we show some retrieval results to illustrate descriptors effectiveness in this heterogeneous databases. As future work, we will evaluate the effectiveness in *yahoo* database using real users.

6.1 Efficiency analysis

Our time measurements were done as follows. Every feature extraction was done three times, and its ellapsed time was calculated: t_1, t_2 and t_3. An arithmetic average was taken from these three calculations, $avg_i^\epsilon = \frac{t_1+t_2+t_3}{3}$,

where i is the image processed. After all images were processed, we took an arithmetic average from all previous averages, $avg^{\epsilon}_{final} = \frac{\sum_{i=1}^{N} avg^{\epsilon}_{i}}{N}$, where N is the number of images in the database. The setup was the same for distance functions time measurements. In the end we have, for each descriptor, one avg^{ϵ}_{final} and one avg^{δ}_{final}. This setup was used to reduce the influence of possible operating system processes preemptions.

Table 3 indicates the relative extraction times considering GCH as reference. We can see that GCH, LCH, and BIC have the fastest extraction algorithms. LCH is 19% slower than GCH and BIC is 55% slower than GCH. CW-HSV is 86% slower than GCH, but is faster than CM, that is almost the double slower than GCH. CCV has the slowest extraction algorithm being almost four times slower than GCH. This means that, when a query image needs to be processed, it will take almost four times longer with CCV than with GCH.

Table 3. Relative extraction times, in the *eth* database.

GCH	BIC	CCV	CM	LCH	CW-HSV
1.00	1.55	3.91	1.98	1.19	1.86

Table 4 shows the same analysis made to distance function times. In this case, the three fastest ones are those related to CW-HSV, CM, and GCH, in this order. CW-HSV is very fast to compare feature vectors, taking almost the half of the time taken by GCH. CM is 17% faster than GCH to compute distances, BIC and CCV are around 40% slower and LCH is almost 7 times slower than GCH.

Table 4. Relative distance function times, in the *eth* database.

GCH	BIC	CCV	CM	LCH	CW-HSV
1.00	1.39	1.40	0.83	6.59	0.56

Table 5 shows the descriptors space requirements for the feature vector of one image. The first row of Table 5 shows the feature vectors size considering the number of (integer) values they contain. The second row shows the absolute feature vectors size in bytes (or bits), considering no compression and that an integer variable has 4 bytes. The third row shows the feature vectors size relative to GCH feature vector (using second row as reference). The number of quantization levels used for all descriptors was 64. This made CCV and BIC generate feature vectors with 128 integer values and GCH generated histograms with 64 values. LCH extraction algorithm partitioned the image in 16 cells (4x4) which made it generate feature vectors with 1024 values. The quantity of moments used for CM descriptor was 6, leading to feature vectors with 12 integer values. CW-HSV generated feature vectors with 63 bits.

Analyzing Table 5 we can see that CW-HSV requires only 3% of the storage required by GCH. CM also generates compact feature vectors, requiring only 19% of the storage required by GCH. BIC and CCV feature vectors have the double size of GCH feature vectors. LCH is far from generating compact feature vectors.

Table 5. Absolute and relative space requirements in *eth* database.

GCH	BIC	CCV	CM	LCH	CW-HSV
64	128	128	12	1024	-
256 B	512 B	512 B	48 B	4096 B	63 bits
1.00	2.00	2.00	0.19	16.00	0.03

6.2 Effectiveness analysis

The effectiveness was analyzed by precision $\times$ recall curves in *eth* database (Figure 1). Precision indicates the quality of the returned results and recall gives the percentage of the relevant images in the database that are presented in the results.

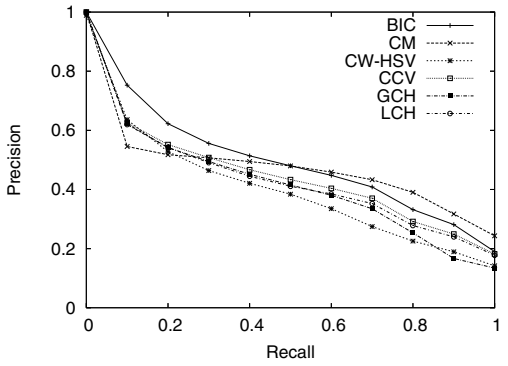

Figure 1. Precision $\times$ Recall curves in *eth* database.

We can see in Figure 1 that BIC presented the highest effectiveness for *recall* values lower than 50%. For recall 10% BIC effectiveness was around 10% higher than all other descriptors. For recall values greater than 50%, CM descriptor had the highest effectiveness. However, for recall values under 20%, CM presented the worst effectiveness. CW-HSV did not presented good effectiveness in *eth* database.

In *yahoo* database we analyzed the retrieval effectiveness based on some query results. Figures 2, 3 and 4 show the first 20 retrieved images for each descriptor for three different queries. The top-left image is the query image.

Figure 2. Query results in *yahoo* database. Top-left image is the query image.

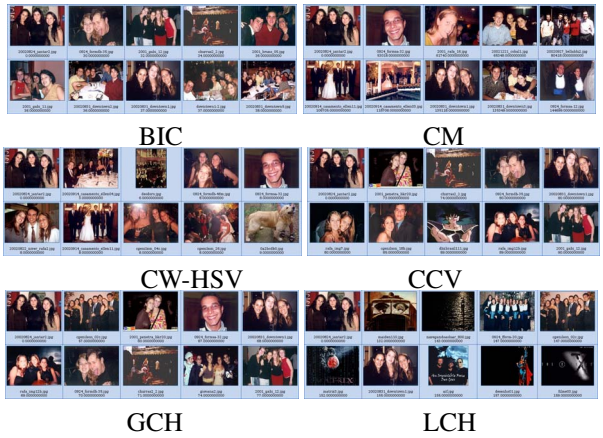

Figure 3. Query results in *yahoo* database. Top-left image is the query image.

Figure 2 has a simple object in the query image. CCV descriptor retrieved similar images in first positions, however, some irrelevant images also appeared. CM descriptor did not presented good effectiveness in this case, retrieving images of people, for example. CW-HSV retrieved relevant images in first positions but also some irrelevant ones were mixed in the results.

Figure 3 has people in the query image. BIC and CCV retrieved very similar images in the first 20 results. All of them containing people. LCH, however, did not retrieved very similar images. CW-HSV presented good results in this case.

Figure 4. Query results in *yahoo* database. Top-left image is the query image.

In Figure 4 the query image is an aerial photo of a city, with lots of buildings. The majority of retrieved images by BIC were images of cities. CM and CW-HSV mixed a lot of images from other categories in the results.

Summarizing, the CM descriptor presented a low effectiveness with our heterogeneous database, while BIC and CCV presented an opposite behavior. CW-HSV did not present good results in some image categories. The effectiveness analysis made here was empirical. A deep analysis will be made using real users, then evaluation measures like precision, for example, will be taken.

7 Conclusion

This paper presented a comparative study of color image descriptors for Web image retrieval. They were compared in terms of extraction algorithm complexity, distance function complexity, space requirements and retrieval effectiveness. The most relevant descriptors were tested in two image databases. One of them with heterogeneous images collected from the Web. Experiments showed that GCH and BIC presented promising results. They generate compact feature vectors, have fast extraction algorithms and perform distance calculations fast. Also, BIC presents good effectiveness for heterogeneous image databases.

As future work, we will use real users to evaluate retrieval effectiveness in *yahoo* database. We will also include more descriptors in our experiments.

8 Acknowledgments

Authors are grateful to FAPESP (process number 2006/59525-1), CAPES, CNPq, and Microsoft Research for financial support.

References

[1] A. E. Abdel-Hakim and A. A. Farag. Csift: A sift descriptor with color invariant characteristics. *cvpr*, 02:1978–1983, 2006.

[2] P. Androutsos, A. Kushki, K. N. Plataniotis, and A. N. Venetsanopoulos. Aggregation of color and shape features for hybrid query generation in content based visual information retrieval. *Signal Process.*, 85(2):385–393, 2005.

[3] R. da S. Torres and A. X. Falcão. Content-Based Image Retrieval: Theory and Applications. *Revista de Informática Teórica e Aplicada*, 13(2):161–185, 2006.

[4] D. da Silva Andrade and N. J. Leite. Testes de significância estatísticos e avaliação de um modelo de recuperação de imagens por conteúdo. Master's thesis, Unicamp, 2004.

[5] R. de Oliveira Stehling and A. X. Falcão. Recuperação por conteúdo em grandes coleções de imagens heterogêneas. Master's thesis, Unicamp, 2002.

[6] E. Hadjidemetriou, M. D. Grossberg, and S. K. Nayar. Multiresolution histograms and their use for recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(7):831–847, 2004.

[7] T. Hurtut, Y. Gousseau, and F. Schmitt. Adptive image retrieval based on the spatial organization of colors. *Computer Vision and Image Understanding*, 2008.

[8] H. Y. Lee, H. K. Lee, and Y. H. Ha. Spatial color descriptor for image retrieval and video segmentation. *Multimedia, IEEE Transactions on*, 5(3):358–367, Sept. 2003.

[9] X. Li. Image retrieval based on perceptive weighted color blocks. *Pattern Recogn. Lett.*, 24(12):1935–1941, 2003.

[10] W.-H. Lin, R. Jin, and A. Hauptmann. Web image retrieval re-ranking with relevance model. In *WI '03: Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence*, page 242, Washington, DC, USA, 2003.

[11] T.-C. Lu and C.-C. Chang. Color image retrieval technique based on color features and image bitmap. *Inf. Process. Manage.*, 43(2):461–472, 2007.

[12] B. Manjunath, J.-R. Ohm, V. Vasudevan, and A. Yamada. Color and texture descriptors. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):703–715, Jun 2001.

[13] D. Messing, P. van Beek, and J. Errico. The mpeg-7 colour structure descriptor: image description using colour and local spatial information. *Image Processing, 2001. Proceedings. 2001 International Conference on*, 1:670–673 vol.1, 2001.

[14] H. A. Moghaddam, T. T. Khajoie, A. H. Rouhi, and M. S. Tarzjan. Wavelet correlogram: A new approach for image indexing and retrieval. *Pattern Recognition*, 38(12):2506–2518, 2005.

[15] K. Nallaperumal, M. S. Banu, and C. C. Christiyana. Content based image indexing and retrieval using color descriptor in wavelet domain. *Conference on Computational Intelligence and Multimedia Applications, 2007. International Conference on*, 3:185–189, 13-15 Dec. 2007.

[16] G. Paschos, I. Radev, and N. Prabakar. Image content-based retrieval using chromaticity moments. *IEEE Transactions on Knowledge and Data Engineering*, 15(5):1069–1072, 2003.

[17] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *MULTIMEDIA '96: Proceedings of the fourth ACM international conference on Multimedia*, pages 65–73, New York, NY, USA, 1996. ACM.

[18] Rajashekhar and S. Chaudhuri. Segmentation and region of interest based image retrieval in low depth of field observations. *Image Vision Comput.*, 25(11):1709–1724, 2007.

[19] R. O. Stehling, M. A. Nascimento, and A. X. Falcao. An adaptive and efficient clustering-based approach for content-based image retrieval in image databases. In *IDEAS '01: Proceedings of the International Database Engineering & Applications Symposium*, pages 356–365, Washington, DC, USA, 2001. IEEE Computer Society.

[20] R. O. Stehling, M. A. Nascimento, and A. X. Falcão. Cell histograms versus color histograms for image representation and retrieval. *Knowl. Inf. Syst.*, 5(3):315–336, 2003.

[21] M. A. Stricker and M. Orengo. Similarity of color images. In W. Niblack and R. C. Jain, editors, *Proc. SPIE Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 381–392, Mar. 1995.

[22] J. Sun, X. Zhang, J. Cui, and L. Zhou. Image retrieval based on color distribution entropy. *Pattern Recognition Letters*, 27(10):1122–1126, 2006.

[23] M. J. Swain and D. H. Ballard. Color indexing. *Int. J. Comput. Vision*, 7(1):11–32, 1991.

[24] L. V. Tran and R. Lenz. Compact colour descriptors for colour-based image retrieval. *Signal Process.*, 85(2):233–246, 2005.

[25] A. Utenpattanant, O. Chitsobhuk, and A. Khawne. Color descriptor for image retrieval in wavelet domain. *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 1:4 pp.–, 20-22 Feb. 2006.

[26] A. Williams and P. Yoon. Content-based image retrieval using joint correlograms. *Multimedia Tools Appl.*, 34(2):239–248, 2007.

[27] K.-M. Wong, L.-M. Po, and K.-W. Cheung. A compact and efficient color descriptor for image retrieval. *Multimedia and Expo, 2007 IEEE International Conference on*, pages 611–614, 2-5 July 2007.

[28] D. Zhang, W. Wang, W. Gao, and S. Jiang. An effective local invariant descriptor combining luminance and color information. *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1507–1510, 2-5 July 2007.